PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-028166
(43)Date of publication of application : 04.02.1994

(51)Int.Cl. G06F 9/06

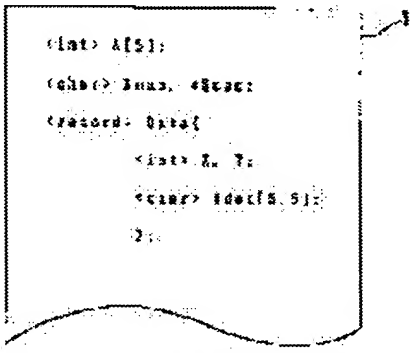
(21)Application number : 04-178516
(22)Date of filing : 06.07.1992

(71)Applicant : TOSHIBA CORP
(72)Inventor : FUJITA TORU

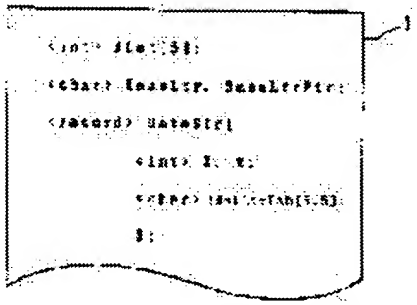
(54) TEXT INPUT/OUTPUT SYSTEM

(57)Abstract:
PURPOSE: To enable a designer to speedily and easily input a program source in his coding style, to easily share design data, and to improve designing efficiency by automatically converting names in the program according to a naming rule.
CONSTITUTION: A variable 'A' (figure A) as an integer type linear array are converted into a name 'ALst' (figure B) according to the naming rule. A character type variable 'Knam' and a character type pointer 'Snam' are converted into 'KnamLtr' and 'SnamLtrPtr' respectively. A record type variable 'Data' is converted into 'DataStr' and the members constituting the record type are also converted according to the naming rule.

(A)



(B)



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号

特開平6－28166

(43)公開日　平成６年(1994)２月４日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/06	４３０　Ｍ	9367－5B		

審査請求　未請求　請求項の数１(全 13 頁)

(21)出願番号　特願平4－178516

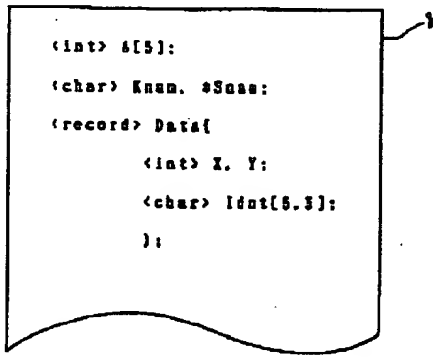
(22)出願日　平成４年(1992)７月６日

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地
(72)発明者　藤田　透
神奈川県川崎市幸区小向東芝町１　株式会社東芝総合研究所内
(74)代理人　弁理士　三好　秀和　(外１名)

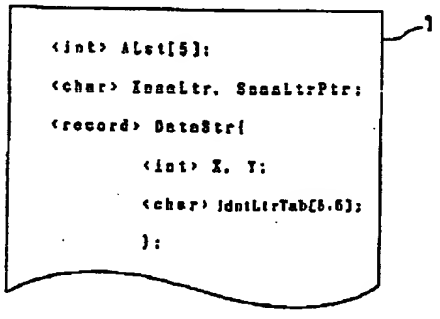
(54)【発明の名称】　テキスト入出力システム

(57)【要約】
【目的】　命名規則に従って自動的にプログラム上の名前の変換を行う。
【構成】　ソースコード１上の、整数型の１次元配列である変数“Ａ”（図（Ａ））は、命名規則に従って“ＡＬｓｔ”という名前に変換される（図（Ｂ））。文字型の変数である“Ｋｎａｍ”、文字型のポインタである“Ｓｎａｍ”はそれぞれ“ＫｎａｍＬｔｒ”，“ＳｎａｍＬｔｒＰｔｒ”という名前に変換される。レコード型の変数である“Ｄａｔａ”は“ＤａｔａＳｔｒ”に変換され、そのレコード型に含まれるメンバーも命名規則に従って名前が変換される。
【効果】　設計者が自らのコーディングスタイルで迅速かつ容易にプログラムソースの入力を行える。設計データの共有化が容易で、設計効率が良くなる。

(A)



(B)



【特許請求の範囲】
【請求項１】　プログラム上で用いられる名前を新たな名前に変換する際の規則を保存している名前変換規則保存手段と、
この名前変換規則保存手段に保存されている名前変換規則に従ってプログラム上の名前を新たな名前に変換する名前変換手段と、
変換される前後の名前を対応付けて格納する名前格納手段とを備えたことを特徴とするテキスト入出力システム。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、プログラム上の名前の変換を自動的に行うことができるテキスト入出力システムに関する。
【０００２】
【従来の技術】プログラム開発などのように複数の設計者が共同作業を行うプロジェクトでは開発者相互のコミュニケーションやソースプログラムのやりとりをスムーズにする必要がある。そのために、プログラム上で用いられるユーザ名、ファイル名、あるいは変数名などの命名規則を設定し、複数の設計者に共通した名前に変換してコーディングを行っていた。
【０００３】
【発明が解決しようとする課題】しかしながら、名前を変換する際、設計者は常に規則を念頭においてコーディングを行わなくてはならないという煩わしさがあった。また、自らのコーディングスタイルと他の設計者のコーディングスタイルとの違いから、作業の効率低下を引き起こすという問題があった。
【０００４】本発明は上述した従来の共同プロジェクトでのコーディング・設計データ管理の欠点を解決するために、プロジェクト管理のために定めた命名規則に従って自動的にプログラム上の名前の変換を行うことによって設計者が従来通りの自らのコーディングスタイルで迅速かつ容易にプログラムソースの入力を行うことができるテキスト入出力システムを提供するものである。
【０００５】
【課題を解決するための手段】上記目的を達成するために、本発明のテキスト入出力システムは、プログラム上で用いられる名前を新たな名前に変換する際の規則を保存している名前変換規則保存手段と、この名前変換規則保存手段に保存されている名前変換規則に従ってプログラム上の名前を新たな名前に変換する名前変換手段と、変換される前後の名前を対応付けて格納する名前格納手段とから構成されている。
【０００６】
【作用】上記構成により、本発明は、名前変換手段が名前変換規則保存手段に保存されている規則を読み込み、プログラム上で名前を見つけると名前変換規則に従ってその名前を変換して名前格納手段に登録する。登録された名前はソースに埋め込まれ、あるいはソースの表示に利用される。
【０００７】
【実施例】以下、この発明の実施例を図面を参照に説明する。
【０００８】まず、この発明をコンパイラに適用した例について図１～４を用いて説明する。
【０００９】図１は、本発明を適用したコンパイラの構成例である。
【００１０】同図において、ソースコード１、字句解析部２、構文解析部３、コード生成部４、保存部５、及び目的コード６は、従来からのコンパイラによる構成である。本発明を適用したコンパイラでは、これらの構成に命名規則指定ファイル７、命名規則指定管理部８、制御部９、名前変換テーブル１０を追加している。
【００１１】命名規則指定ファイル７は、変換規則を保存しているファイルであり、詳細は後述する。
【００１２】命名規則指定管理部８は、命名規則指定ファイル７に保存されている規則に従って名前を変換するところである。
【００１３】制御部９は、命名規則指定管理部８などを制御するものである。
【００１４】名前変換テーブル１０は、変換された前後の名前を対応付けて格納するところである。
【００１５】図２は、設計者によって作成された命名規則指定ファイル７の具体例である。この例では使用するプログラム言語に整数型、文字型、及び複数のデータ型のセットであるレコード型の３つのデータ型を仮定して、それぞれのデータ型に配列やポインタの形式を指定することができるとする。
【００１６】この命名規則指定ファイル７は、これらのデータ型、データ形式に右欄の文字列を付加させる指定を行うものである。この例では複数型の変数は変更なし、文字型・レコード型の変数はそれぞれ“Ｌｔｒ”，“Ｓｔｒ”という文字列をその名前の後につけ加える。さらに変数のデータ形式によって１次元・２次元の配列にはそれぞれ“Ｌｓｔ”，“Ｔａｂ”、ポインタには“Ｐｔｒ”を名前の後につけ加える。
【００１７】このように、この発明を適用したコンパイラは構成されており、次にこのコンパイラの動作を説明する。
【００１８】字句解析部２がプログラムのソースコード１を字句単位で読み込む。読み込まれた字句のうちでプログラム言語の仕様により名前（例えば、定数名、関数名、変数名など）であると判断されたものは命名規則指定管理部８に渡されて命名規則に従って変換され、命名規則指定管理部８から字句解析部２に返される。
【００１９】図３は、読み込まれ変換される前後のソースコードの例である。図（Ａ）が変換前のソースコー

ド、（Ｂ）が変換後のソースコードである。この例では整数型の１次元配列である変数“Ａ”は命名規則に従って“ＡＬｓｔ”という名前に変換されている。以下同様にして文字型の変数である“Ｋｎａｍ”、文字型のポインタである“Ｓｎａｍ”はそれぞれ“ＫｎａｍＬｔｒ”，“ＳｎａｍＬｔｒＰｔｒ”という名前に変換される。レコード型の変数である“Ｄａｔａ”は“ＤａｔａＳｔｒ”に変換され、そのレコード型に含まれるメンバーも命名規則に従って名前が変換される。

【００２０】また命名規則指定管理部８は、変換前後の名前を名前変換テーブル１０に登録する。図４は名前変換テーブルの例である。この例では変換前後の変数名・変数のデータ型とデータ形式を保存している。このテーブルの作成後は、ソースコードに現れる名前の変換はこのテーブルを参照して行なわれる。

【００２１】さらに、名前が変換されたソースコード１が構文解析部３・コード生成部４・保存部５と順番に渡されて目的コード６がファイルとして保存される。また制御部９によって名前変換後のソースコード１がファイルに保存される。

【００２２】２番目の例として本発明が設計データ管理に適用された例を示す。

【００２３】図５は複数の設計者が使用するデータ管理システムの構成例である。

【００２４】図５においては、ソースコード１、データ読み込み部１１、データ表示部１２、データ保存部１３、保存ファイル１４が従来からの構成であり、ファイル名設定ファイル１５、命名規則指定管理部８、制御部９、名前変換テーブル１０がこの発明によって追加された機能である。

【００２５】命名規則指定管理部８、制御部９、名前変換テーブル１０は、図１で示したものと同様の機能を有している。ファイル名設定ファイル１５は、図１の命名規則指定ファイル７に相当するものであり、ファイル名の変換規則を保存しているファイルである。

【００２６】このように構成されたデータ管理システムでは、データ読み込み部１１が、ある名前のファイルに保存されているソースコード１を読み込むと同時に、制御部９がユーザの名前を確認してファイル名設定ファイル１５の指定する規則にしたがってファイル名を変換する。

【００２７】図６はファイル名設定ファイルの例である。図６のはじめの文字列“ＰＲＯＡ”はプロジェクト名である。次の数字はそのプロジェクトに関わるグループの数である。その後は各グループの担当するサブプロジェクト名、グループメンバーの数とグループメンバーの名前である。

【００２８】いまサブプロジェクト“ＣＰＵ”に携わっている“ｆｕｊｉｔａ”というメンバーが“ＡＢＵＳ”というファイルを新たに作成すると、データ管理システムはこのファイルのソースコード１を読み込み、ファイル名設定ファイル１５を参照してファイル名を“ＰＲＯＡ＿ＣＰＵ＿ＡＢＵＳ＿ｆｕｊｉｔａ”に変換する。そして、変換前後のファイル名を名前変換テーブル１０に登録する。

【００２９】これにより、他のメンバーがこのファイルをデータ管理システムから利用しようとする場合には、この変更後の名前で参照することができる。ファイルの作成者である“ｆｕｊｉｔａ”がこのファイルを利用しようとする場合には、制御部９がメンバー名をチェックしてファイル名設定ファイル１５を参照し、変更前後のどちらの名前でも参照できる。

【００３０】また、データ読み込み部１１が名前を読み込んだ時にそれを命名規則指定管理部８に渡して、名前変換テーブル１０を利用して変換前の名前に戻して表示することもできる。

【００３１】ソースコード１に修正が加わった場合には、制御部９が修正されたことを確認し、データ読み込み部１１から必要なデータを受けとって名前変更テーブル１０の修正を行なう。このため設計者は、自分のコーディングスタイルで作成したソースコード１のままであると仮定してデータを利用したり、変更することができる。

【００３２】３番目の例として本発明がハードウェア設計フレームワークに適用された例を示す。

【００３３】図７は、ハードウェア設計フレームワークに本発明を適用したシステムの構成例である。

【００３４】図７で示したシステムにおいては、フレームワークシステム１６、データ変換部１７が従来から備えられている構成であり、命名規則指定管理部８、制御部９、名前変換テーブル１０、及びモジュール階層データ１８が新たに追加構成されたものである。

【００３５】図５の説明の際にも述べたように、命名規則指定管理部８、制御部９、名前変換テーブル１０は、図１で示したものと同様の機能を有している。モジュール階層データ１８は、図１の命名規則指定ファイル７や図５のファイル名設定ファイル１５に相当するものであり、モジュール名の変換規則を保存しているデータファイルである。

【００３６】このように構成されたハードウェア設計フレームワークシステムの作用を以下で説明する。

【００３７】図８はフレームワークでの設計画面の例である。図８の“ＳＹＳＡ”というシートは一つのモジュールを表しており、モジュール“ＳＹＳＡ”はその中にモジュール“Ａ”，“Ｂ”，“ＭｕｌｔＰ２”などを含んでいる。

【００３８】いまモジュール“ＳＹＳＡ”の設計者がモジュール“Ｂ”の内部構造を設計し、新たにシート“Ｂ”を作成する。その際にシート“ＳＹＳＡ”に含まれているモジュール“ＭｕｌｔＰ２”をシート“Ｂ”の

内部にコピーして使用している。この時点ではモジュール“ＭｕｌｔＰ２”を表現する設計データは一つしか存在していない。

【００３９】フレームワークシステム１６は、これらのデータからモジュール階層データ１８を抽出する。制御部９は新しいシートが作成されたことが判るとモジュール階層データ１８を命名規則指定管理部８に読み込ませる。そしてモジュール階層データ１８に基づいて図９のようにモジュール名を変換して名前変換テーブル１０を作成する。

【００４０】図１０は名前変換テーブル１０の例である。テーブル１（図（Ａ））を見ると、モジュール“ＳＹＳＡ”の下位モジュールである“Ａ”，“Ｂ”，“Ｃ”，“ＭｕｌｔＰ２”はそれぞれ“ＳＹＳＡ＿Ａ”，“ＳＹＳＡ＿Ｂ”，“ＳＹＳＡ＿Ｃ”，“ＳＹＳＡ＿ＭｕｌｔＰ２”に変換されている。モジュール“ＳＹＳＡ＿Ｂ”の下位モジュールである“ＭｕｌｔＰ２”は“ＳＹＳＡ＿Ｂ＿ＭｕｌｔＰ２”に変換されている。

【００４１】設計者が自らのコーディングスタイルに従って自由にモジュール名（ここでは、“Ａ”，“Ｂ”，“Ｃ”，“ＭｕｌｔＰ２”）をつけると、自動的にモジュールの階層構造を反映したモジュール名に変換され、ほかの設計者とのデータの共有が容易になる。

【００４２】ここで同じ構造を持つモジュールである“ＳＹＳＡ＿ＭｕｌｔＰ２”と“ＳＹＳＡ＿Ｂ＿ＭｕｌｔＰ２”に別の名前がついているが、モジュールの設計データは“ＭｕｌｔＰ２”が一つだけ存在していて、フレームワークシステム１６が名前変換テーブル１０を参照してそれぞれの名前にモジュール設計データを対応させている。

【００４３】もしここでモジュール“ＳＹＳＡ＿Ｂ＿ＭｕｌｔＰ２”の設計データを変更しようとすると、制御部９は命名規則指定管理部８に知らせて名前変換テーブル１０から、同じ設計データを持つモジュール名“ＳＹＳＡ＿Ｂ＿ＭｕｌｔＰ２”，“ＳＹＳＡ＿ＭｕｌｔＰ２”を取り出し、設計者にデータの変更をどのモジュールに対して行うかを指定させる。このとき、同じ設計データを持つすべてのモジュールを指定すれば、データの変更はただ一つの設計データである“ＭｕｌｔＰ２”に対して行われる。

【００４４】またモジュール“ＳＹＳＡ＿Ｂ＿ＭｕｌｔＰ２”のみを指定した場合には設計データ“ＭｕｌｔＰ２”に変更の加えられた結果が新たに設計データ“ＭｕｌｔＰ２＿２”として保存され、名前変換テーブル１０は図１０（Ｂ）のテーブル２のように変更される。このためモジュール設計データ量の増大を防ぐことができる。

【００４５】

【発明の効果】以上述べてきたように、本発明によれば、設計者は、自らのコーディングスタイルを変更することなしに設計データの作成を行なうことができるため、煩わしさを軽減できるとともに、設計データの共有化が容易となり、効率の良い設計が可能となる。

【図面の簡単な説明】

【図１】本発明をコンパイラに適用した構成例。

【図２】図１で示した命名規則指定ファイルの具体例。

【図３】図１で示したコンパイラによって変換される前後のソースコードの例。

【図４】図１で示した名前変換テーブルの例。

【図５】本発明をデータ管理システムに適用した構成例。

【図６】図５で示したファイル名設定ファイルの例。

【図７】本発明をフレームワークに適用した構成例。

【図８】図７で示したフレームワークで変換される前のモジュール名。

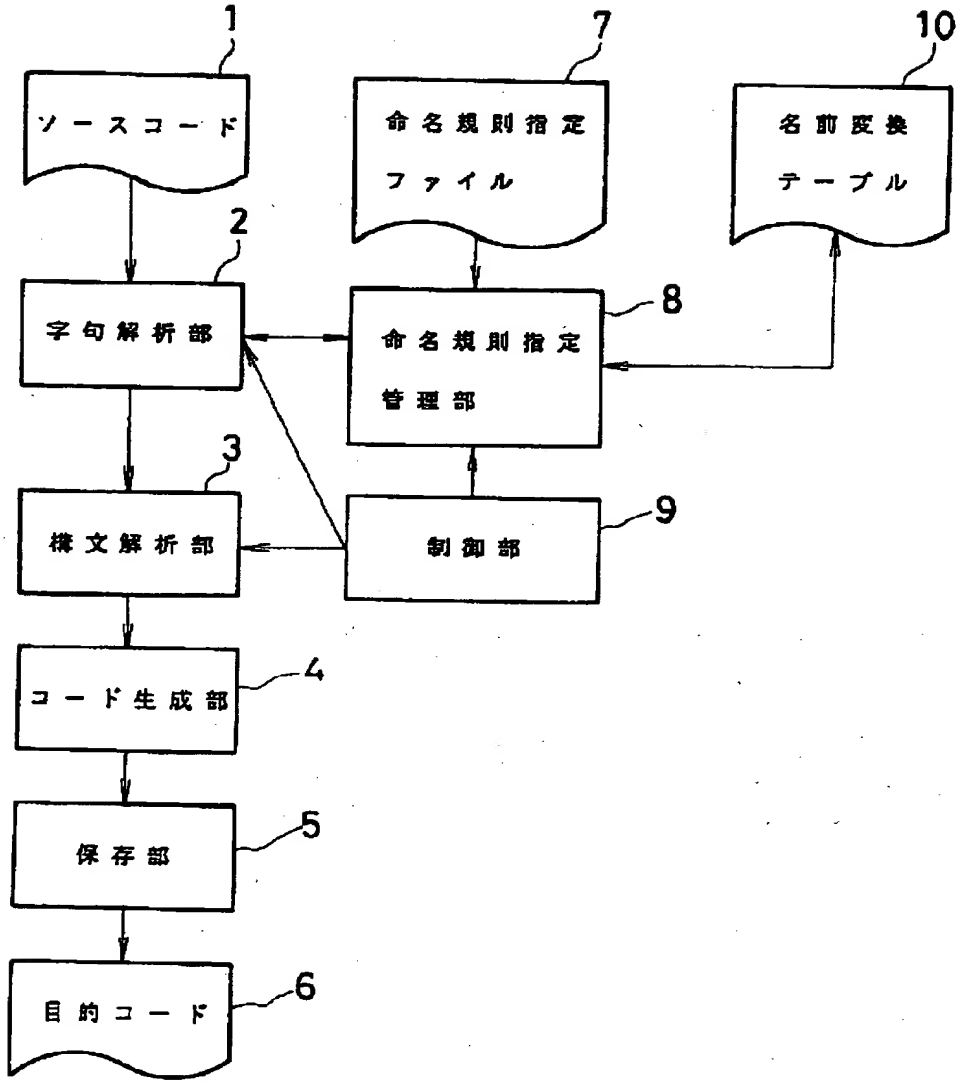
【図９】図７で示したフレームワークで変換された後のモジュール名。

【図１０】図７で示した名前変換テーブルの例。

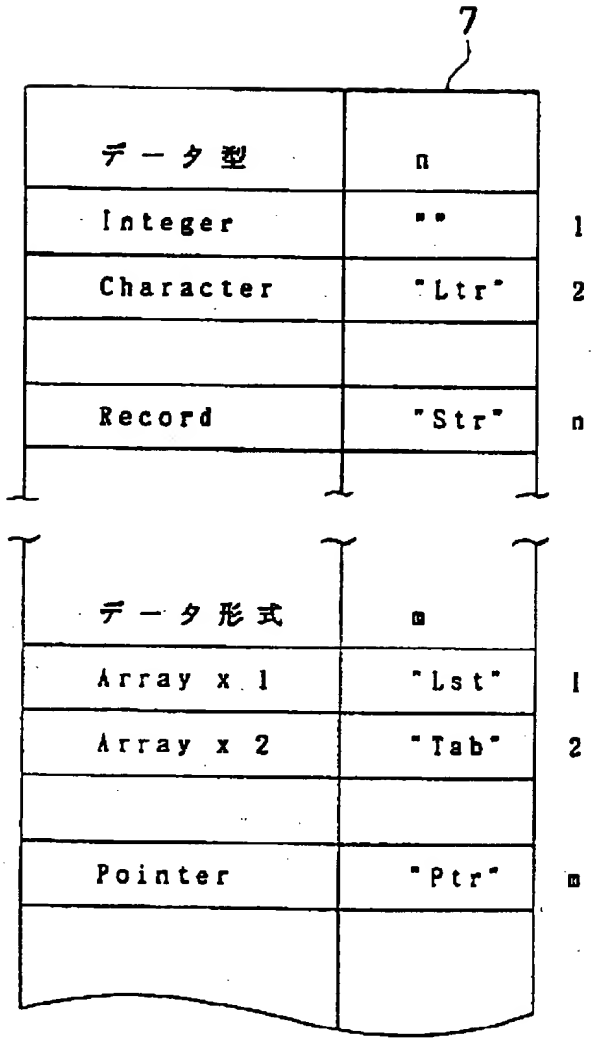
【符号の説明】

７　命名規則指定ファイル
８　命名規則指定管理部
９　制御部
１０　名前変換テーブル
１５　ファイル名設定ファイル
１８　モジュール階層データ

【図１】


1
ソースコード
7
命名規則指定ファイル
10
名前変換テーブル
2
字句解析部
8
命名規則指定管理部
3
構文解析部
9
制御部
4
コード生成部
5
保存部
6
目的コード

【図２】


7
データ型
n
Integer
""
1
Character
"Ltr"
2
Record
"Str"
n
データ形式
m
Array x 1
"Lst"
1
Array x 2
"Tab"
2
Pointer
"Ptr"
m

【図３】

(A)

```
<int> A[5];
<char> Knam, *Snam;
<record> Data{
        <int> X, Y;
        <char> Idnt[5,5];
        };
```

1

(B)

```
<int> ALst[5];
<char> KnamLtr, SnamLtrPtr;
<record> DataStr{
        <int> X, Y;
        <char> IdntLtrTab[5,5];
        };
```

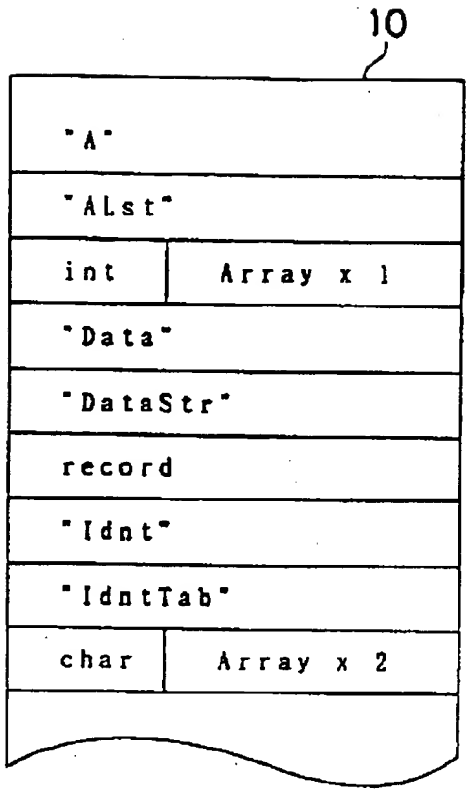
1

【図６】

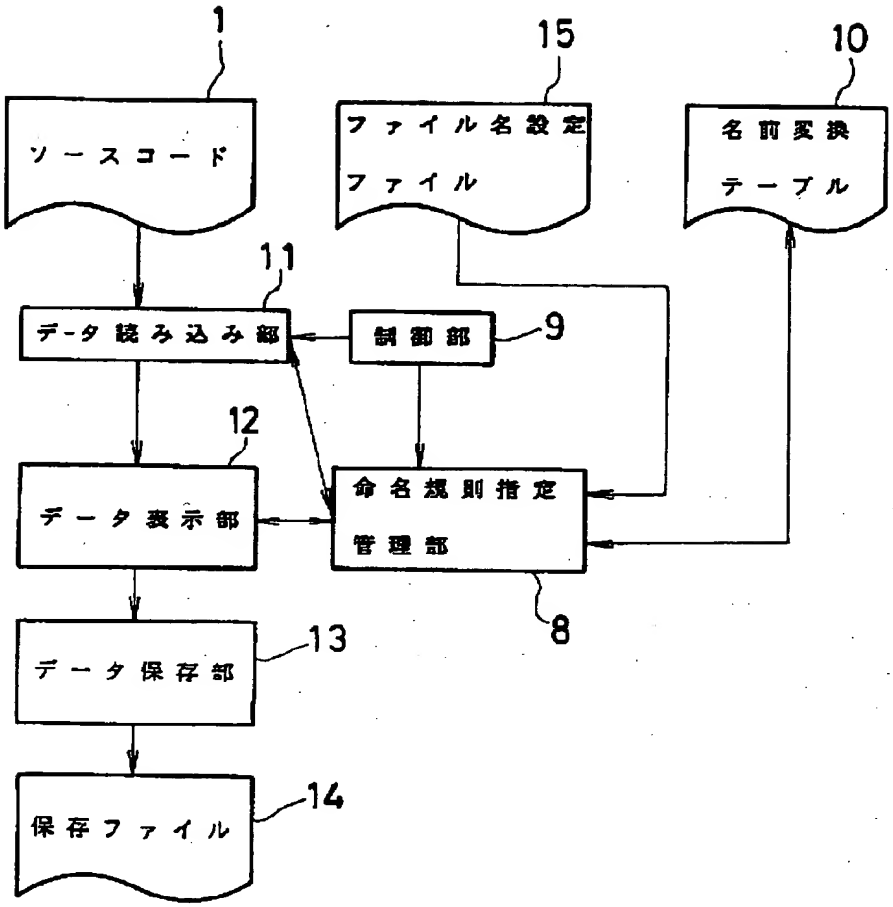
15

"PROA"
n
"CPU"
2
"fujita"
"sakai"
"BUS"
1
"takahashi"

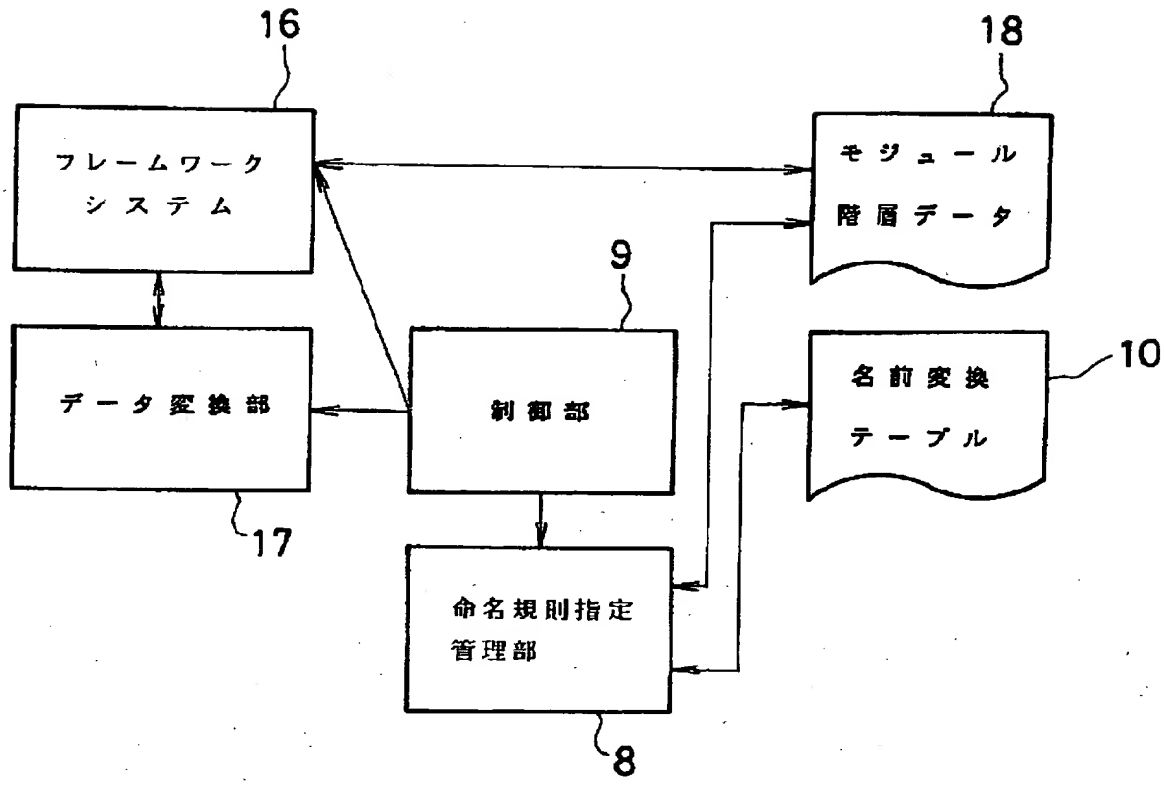
【図４】


10
"A"
"ALst"
int
Array x 1
"Data"
"DataStr"
record
"Idnt"
"IdntTab"
char
Array x 2

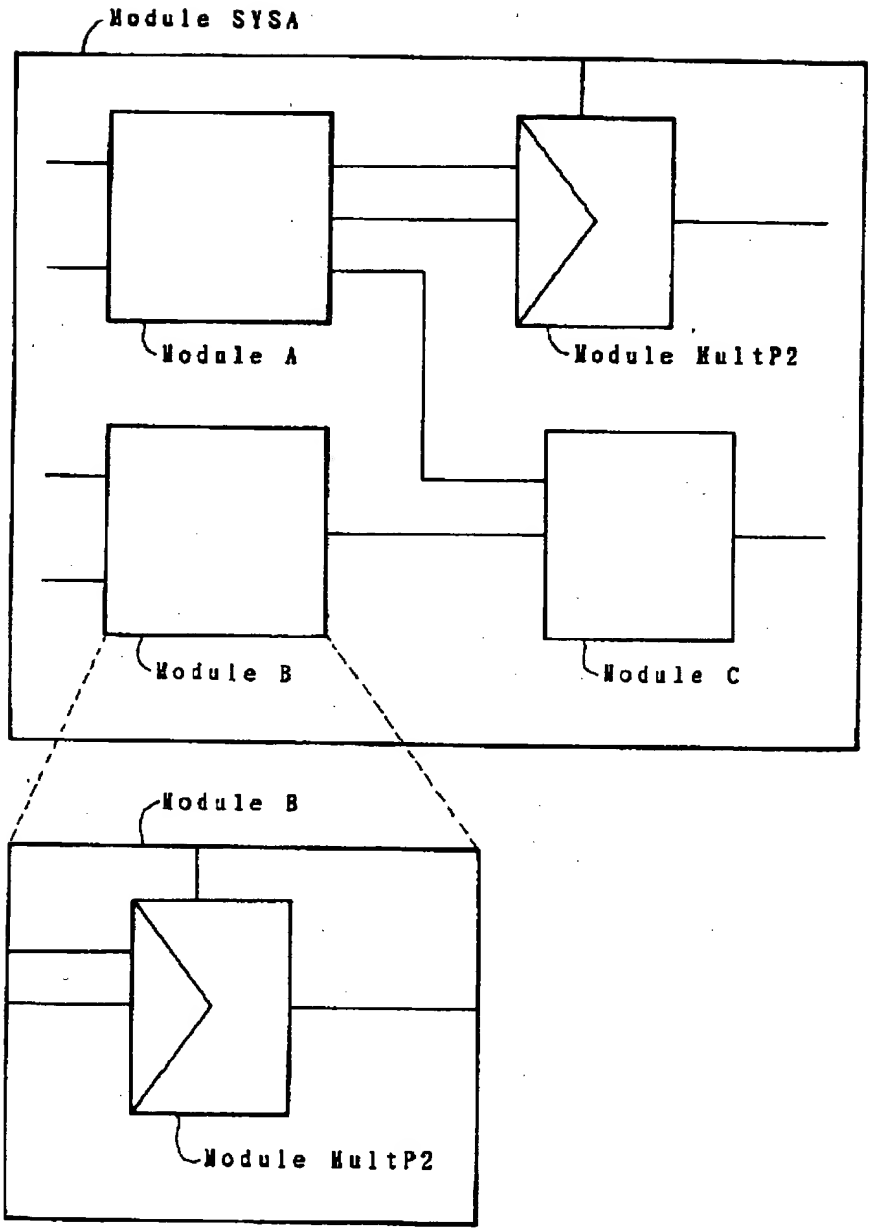
【図５】


1
ソースコード
15
ファイル名設定ファイル
10
名前変換テーブル
11
データ読み込み部
制御部
9
12
データ表示部
命名規則指定管理部
8
データ保存部
13
保存ファイル
14

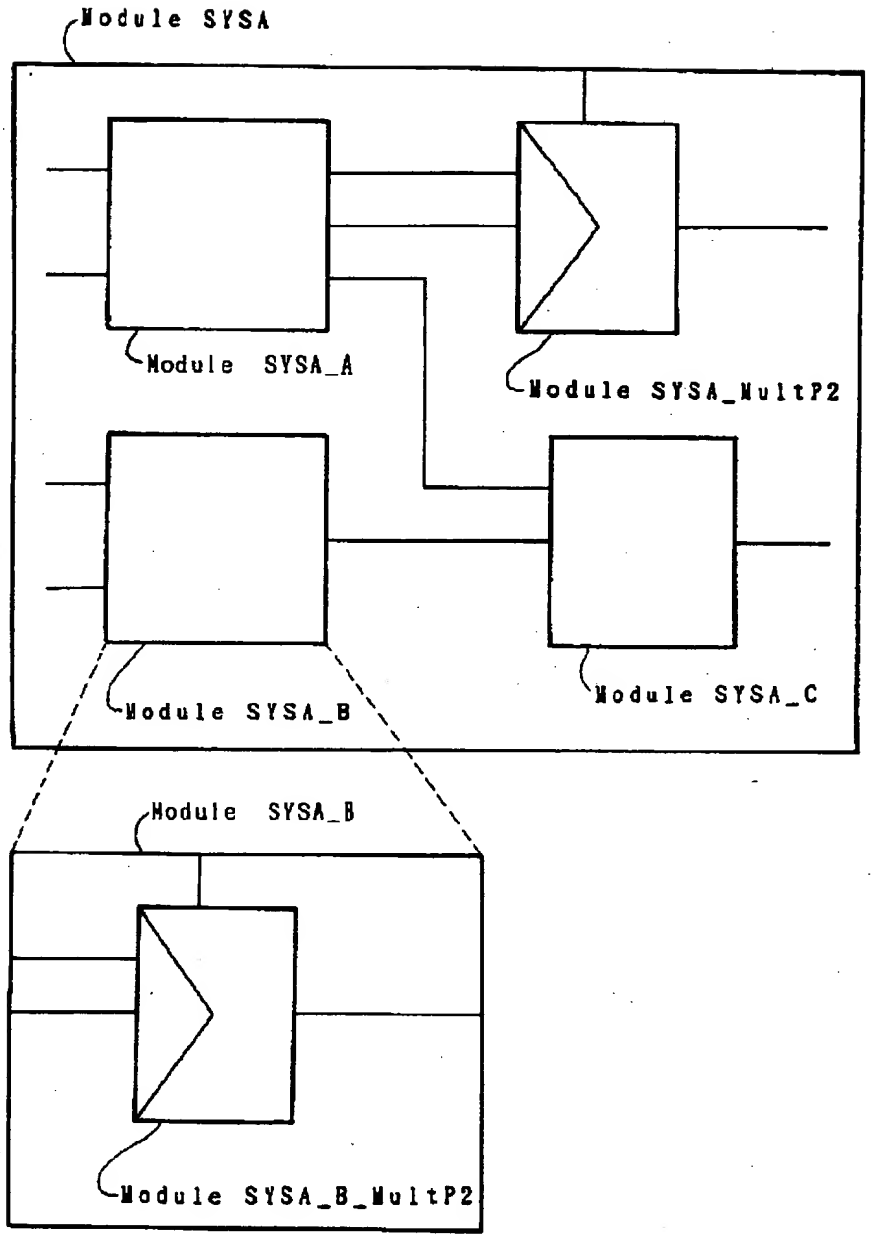
【図７】


16
フレームワーク
システム
18
モジュール
階層データ
9
制御部
データ変換部
17
名前変換
テーブル
10
命名規則指定
管理部
8

【図８】


Module SYSA
Module A
Module MultP2
Module B
Module C
Module B
Module MultP2

【図９】


Module SYSA
Module SYSA_A
Module SYSA_MultP2
Module SYSA_B
Module SYSA_C
Module SYSA_B
Module SYSA_B_MultP2

【図１０】

(A)

SYSA	1
SYSA	
A	1
SYSA_A	
B	1
SYSA_B	
C	1
SYSA_C	
MultP2	2
SYSA_MultP2	
SYSA_B_MultP2	

テーブル１

(B)

SYSA	1
SYSA	
A	1
SYSA_A	
B	1
SYSA_B	
C	1
SYSA_C	
MultP2	1
SYSA_MultP2	
MultP2_2	1
SYSA_B_MultP2	

テーブル２